FIG. 1A

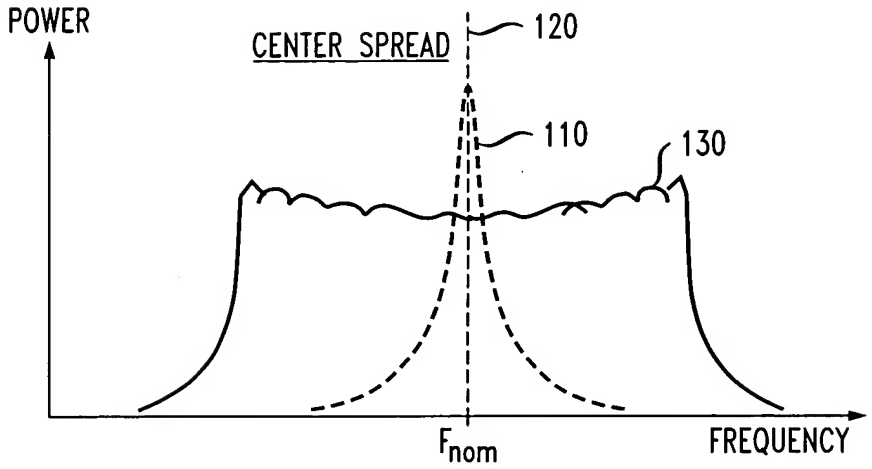


FIG. 1B

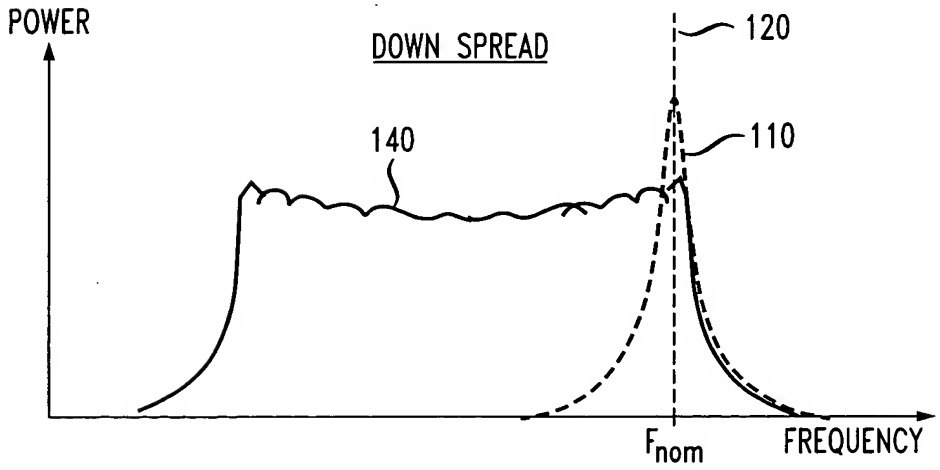


FIG. 2

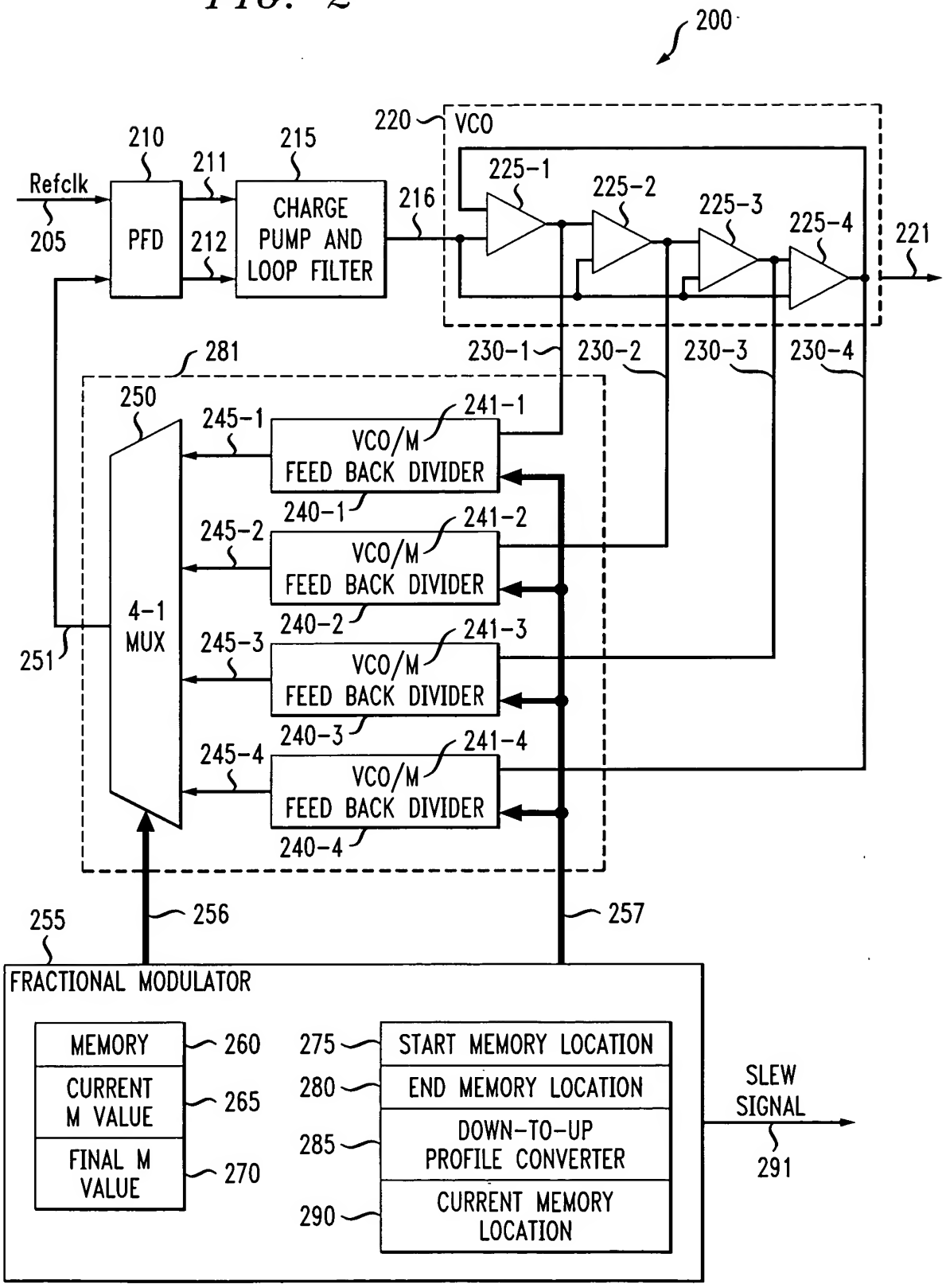

200
220
VCO
210
211
215
Refclk
205
PFD
212
CHARGE PUMP AND LOOP FILTER
216
225-1
225-2
225-3
225-4
221
281
230-1
230-2
230-3
230-4
250
245-1
241-1
VCO/M FEED BACK DIVIDER
240-1
245-2
241-2
VCO/M FEED BACK DIVIDER
240-2
4-1 MUX
251
245-3
241-3
VCO/M FEED BACK DIVIDER
240-3
245-4
241-4
VCO/M FEED BACK DIVIDER
240-4
255
256
257
FRACTIONAL MODULATOR
MEMORY
260
CURRENT M VALUE
265
FINAL M VALUE
270
275
START MEMORY LOCATION
280
END MEMORY LOCATION
285
DOWN-TO-UP PROFILE CONVERTER
290
CURRENT MEMORY LOCATION
SLEW SIGNAL
291

FIG. 3B

	313 M	315 PHASE
317-1	50 (0)	1
317-2	49 (-1)	4
317-3	50 (0)	3
317-4	50 (0)	2
317-5	49 (-1)	4
317-6	50 (0)	2
317-7	49 (-1)	4
317-8	50 (0)	2
317-9	49 (-1)	3
317-10	50 (0)	4
317-11	50 (0)	1
317-12	49 (-1)	2
317-13	50 (0)	2
317-14	49 (-1)	2
317-15	50 (0)	2
317-16	49 (-1)	2

300

311 321 331 341

FIG. 3A

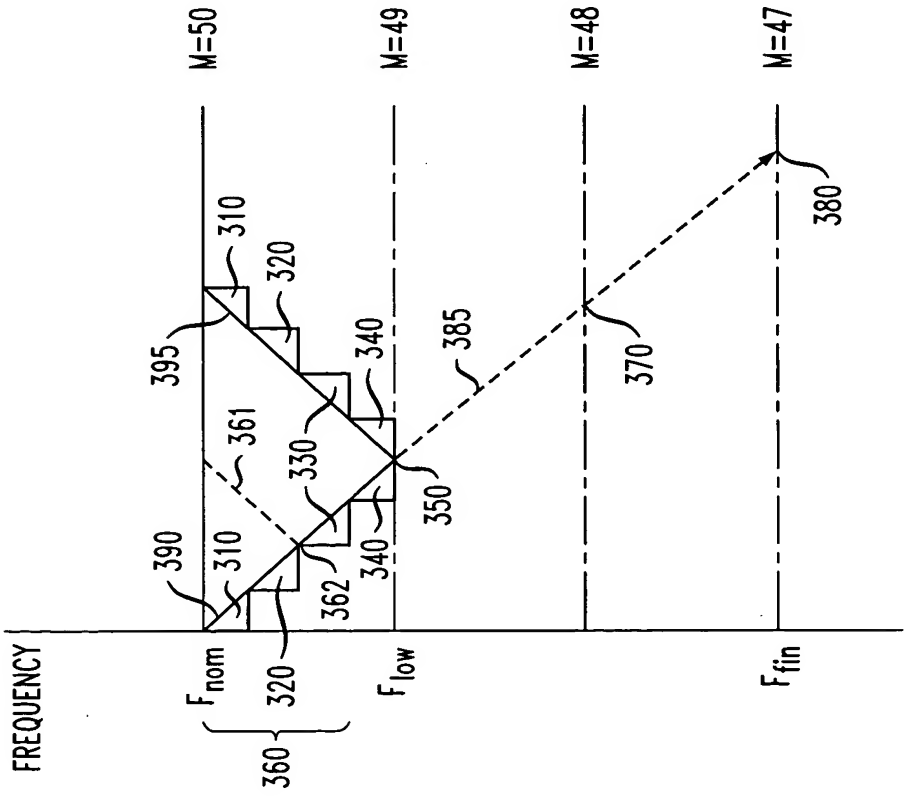


FIG. 3C

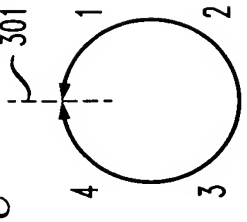

301
1
2
3
4

FIG. 4

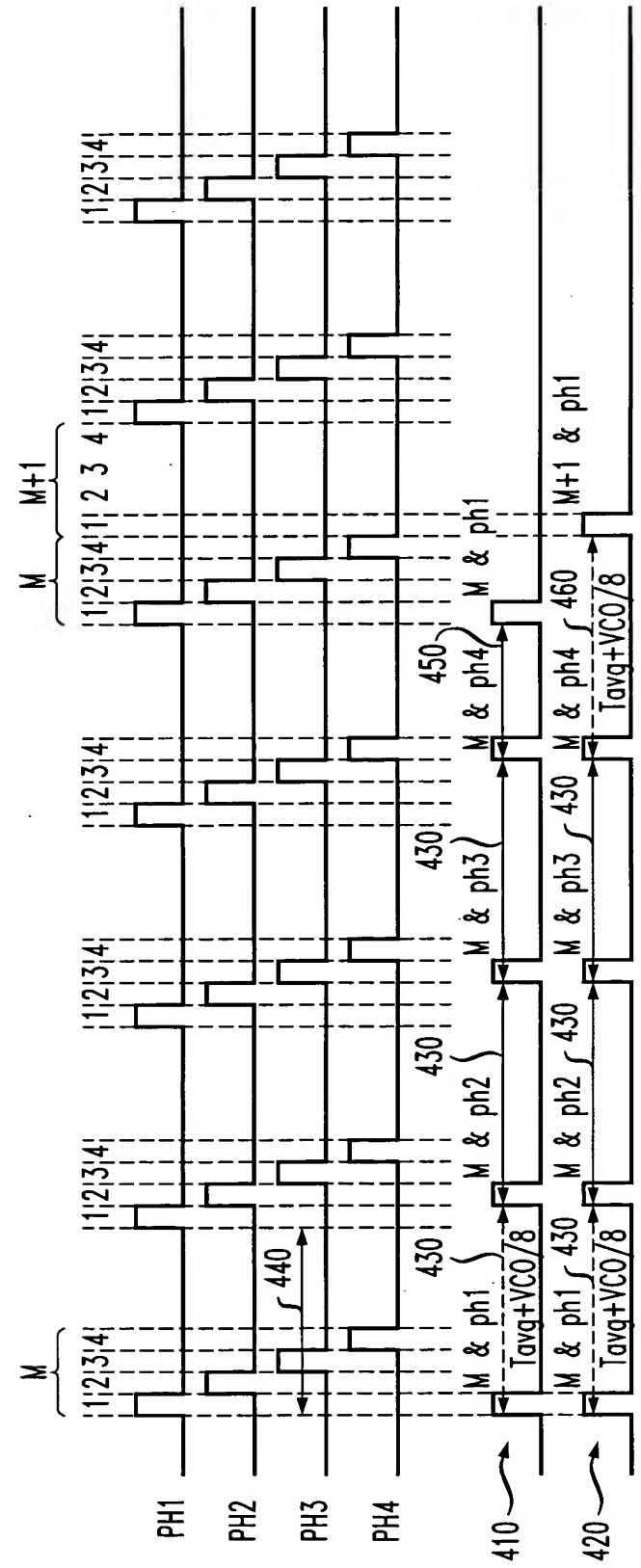

M
M+1
PH1
PH2
PH3
PH4
410
420
440
430
450
460
M & ph1
M & ph2
M & ph3
M & ph4
Tavg+VCO/8
M+1 & ph1

FIG. 5

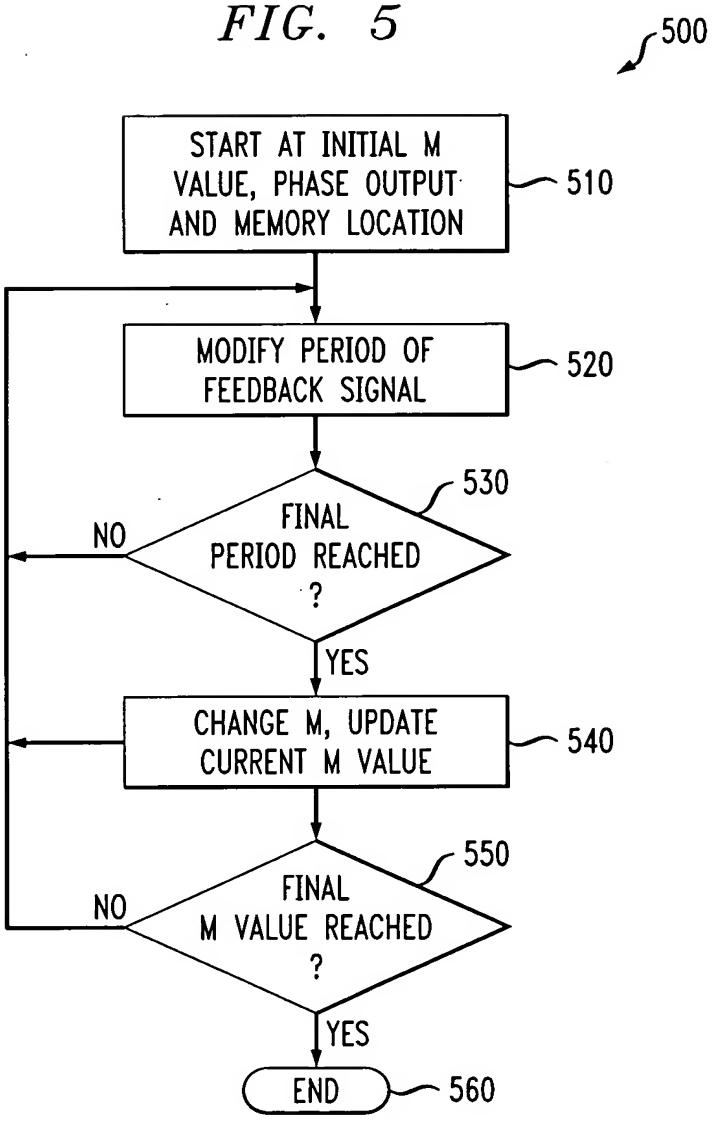

500
START AT INITIAL M VALUE, PHASE OUTPUT AND MEMORY LOCATION
510
MODIFY PERIOD OF FEEDBACK SIGNAL
520
530
FINAL PERIOD REACHED ?
NO
YES
CHANGE M, UPDATE CURRENT M VALUE
540
550
FINAL M VALUE REACHED ?
NO
YES
END
560

FIG. 6

	FRACTIONAL M
610	50
	49.875
	50
	50
620	50
	49.875
	49.875
	50
630	50
	49.875
	49.875
	49.875
640	49.875
	49.875
	49.875

FIG. 7

700

M	PHASE
50 (0)	1
49 (-1)	8
50 (0)	7
50 (0)	6
50 (0)	5
50 (0)	4
50 (0)	3
50 (0)	2
50 (0)	1
49 (-1)	8

710

M	PHASE	
50 (0)	1	717-1
50 (0)	2	717-2
50 (0)	3	717-3
50 (0)	4	717-4
50 (0)	5	717-5
50 (0)	6	717-6
50 (0)	7	717-7
50 (0)	8	717-8
51 (+1)	1	717-9
50 (0)	2	717-10

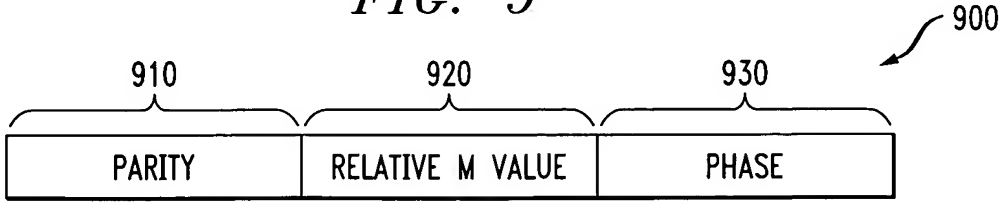
FIG. 8


```
module dn_to_up_profile      ( // Inputs
                               slew_up, PII_Clk_Early, RESET,
                               IN_MOFSET2, IN_MOFSET1, IN_MOFSET0,
                               IN_SELPH3, IN_SELPH2, IN_SELPH1, IN_SELPH0,
                               // Outputs
                               OUT_MOFSET2, OUT_MOFSET1, OUT_MOFSET0,
                               OUT_SELPH3, OUT_SELPH2, OUT_SELPH1, OUT_SELPH0
                             );
input       slew up;          // Active high control. Outputs=inputs if slew_up=0.
input       PII_Clk_Early;    // Clock.
input       RESET;            // Active high reset.
input       IN_MOFSET2;       // The M bits.
input       IN_MOFSET1;
input       IN_MOFSET0;
input       IN_SELPH3;        // Expect always 0 so not sampled.
input       IN_SELPH2;        // The phase bits.
input       IN_SELPH1;
input       IN_SELPH0;

output      OUT_MOFSET2;      // The M bits.
output      OUT_MOFSET1;
output      OUT_MOFSET0;
output      OUT_SELPH3;       // Always 0.
output      OUT_SELPH2;       // The phase bits.
output      OUT_SELPH1;
output      OUT_SELPH0;

reg [2:0]   last_selph;
reg [2:0]   local_selph;
wire[2:0]   current_selph;

always @( IN_SELPH2 or IN_SELPH1 or IN-SELPH0 )
   begin
          // 2s-complement implementation
          case ( {IN_SELPH2, IN_SELPH1, IN_SELPH0} )
             3'b000 : local_selph = 3'b000;
             3'b001 : local_selph = 3'b111;
             3'b010 : local_selph = 3'b110;
             3'b011 : local_selph = 3'b101;
             3'b100 : local_selph = 3'b100;
             3'b101 : local_selph = 3'b011;
             3'b110 : local_selph = 3'b010;
             3'b111 : local_selph = 3'b001;
          endcase // case( {IN_SELPH2, IN_SELPH1, IN_SELPH0} )
   end // always @ ( IN-SELPH2 or IN_SELPH1 or IN_SELPH0 )
```

FIG. 8 cont.

```
assign      current_selph[2] = slew_up ? local_selph[2] : IN_SELPH2;
assign      current_selph[1] = slew_up ? local_selph[1] : IN_SELPH1;
assign      current_selph[0] = slew_up ? local_selph[0] : IN_SELPH0;

always @( posedge RESET or posedge PII_Clk_Early)
    begin
            if ( RESET == 1'b1 )
                begin
                    last_selph <= 4'b0000;
                end
            else
                begin
                    last_selph <= current_selph;
                end

    end // always @ ( posedge RESET or posedge PII_Clk_Early)

assign      OUT_SELPH3 = slew_up ? 1'b0 : IN_SELPH3;
assign      OUT_SELPH2 = slew_up ? local_selph[2] : IN_SELPH2;
assign      OUT_SELPH1 = slew_up ? local_selph[1] : IN_SELPH1;
assign      OUT_SLEPH0 = slew_up ? local_selph[0] : IN_SELPH0;

// MOFSET for slewing up will be either 000 or 001, depending
// on whether the phase has just rolled over from 7 to 0.
assign      OUT_MOFSET2 = slew_up ? 1'b0 : IN_MOFSET2;
assign      OUT_MOFSET1 = slew_up ? 1'b0 : IN_MOFSET1;
assign      OUT_MOFSET0 = slew_up ? ( ( (last_selph > local_selph) ||
                                ((last_selph == local_selph) &&
                                 (IN_MOFSET2 == 3'b1) &&
                                 (IN_MOFSET1 == 3'b0) &&
                                 (IN_MOFSET0 == 3'b1) ) )
                                          ? 1'b1 : 1'b0 ) : IN_MOFSET0;
endmodule
```

FIG. 9



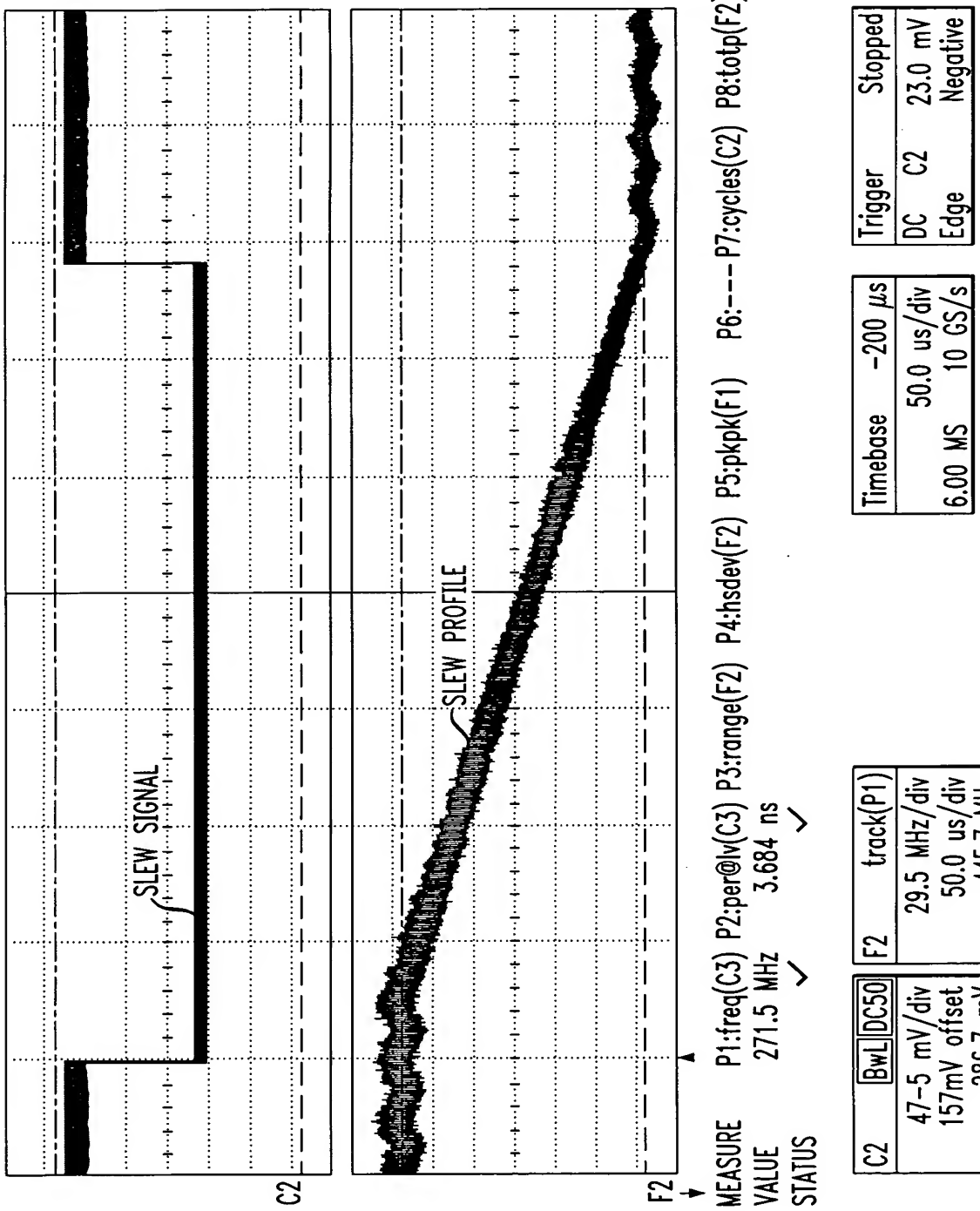

SLEW SIGNAL
SLEW PROFILE
C2
F2
MEASURE
VALUE
STATUS
P1:freq(C3) P2:per@lv(C3) P3:range(F2) P4:hsdev(F2) P5:pkpk(F1) P6:--- P7:cycles(C2) P8:totp(F2)
271.5 MHz 3.684 ns
C2 BwL DC50
47-5 mV/div
157mV offset
286.7 mV
3.1 mV
F2 track(P1)
29.5 MHz/div
50.0 us/div
445.7 MHz
289.6 MHz
Timebase -200 µs
50.0 us/div
6.00 MS 10 GS/s
Trigger Stopped
DC C2 23.0 mV
Edge Negative

FIG. 10